Unit 1. Basic Notions

1.1 Computer Science and Engineering
1.2 Basic Concepts
1.3 Information representation
1.4 Programming Languages
1.5 Compilers

Objectives

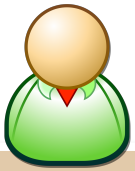
- Understand which the basic components of a computer are,
- Understand how the data is stored,
- Understand what a programming language is

Computer Science and Engineering

Definitions:

- Computer science, or computing science, is:
  - the study of the theoretical foundations of information and computation, and
  - their implementation and application in computer systems.

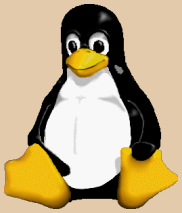
Basic concepts: Computer Architecture



User

Application Software

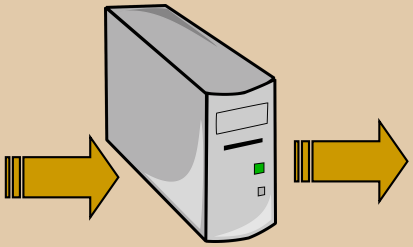
Operating Systems



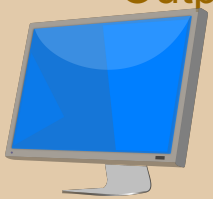
Hardware

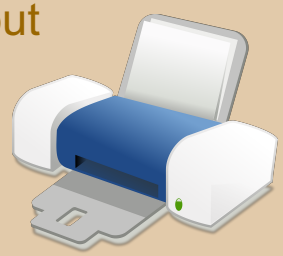
Input

Process



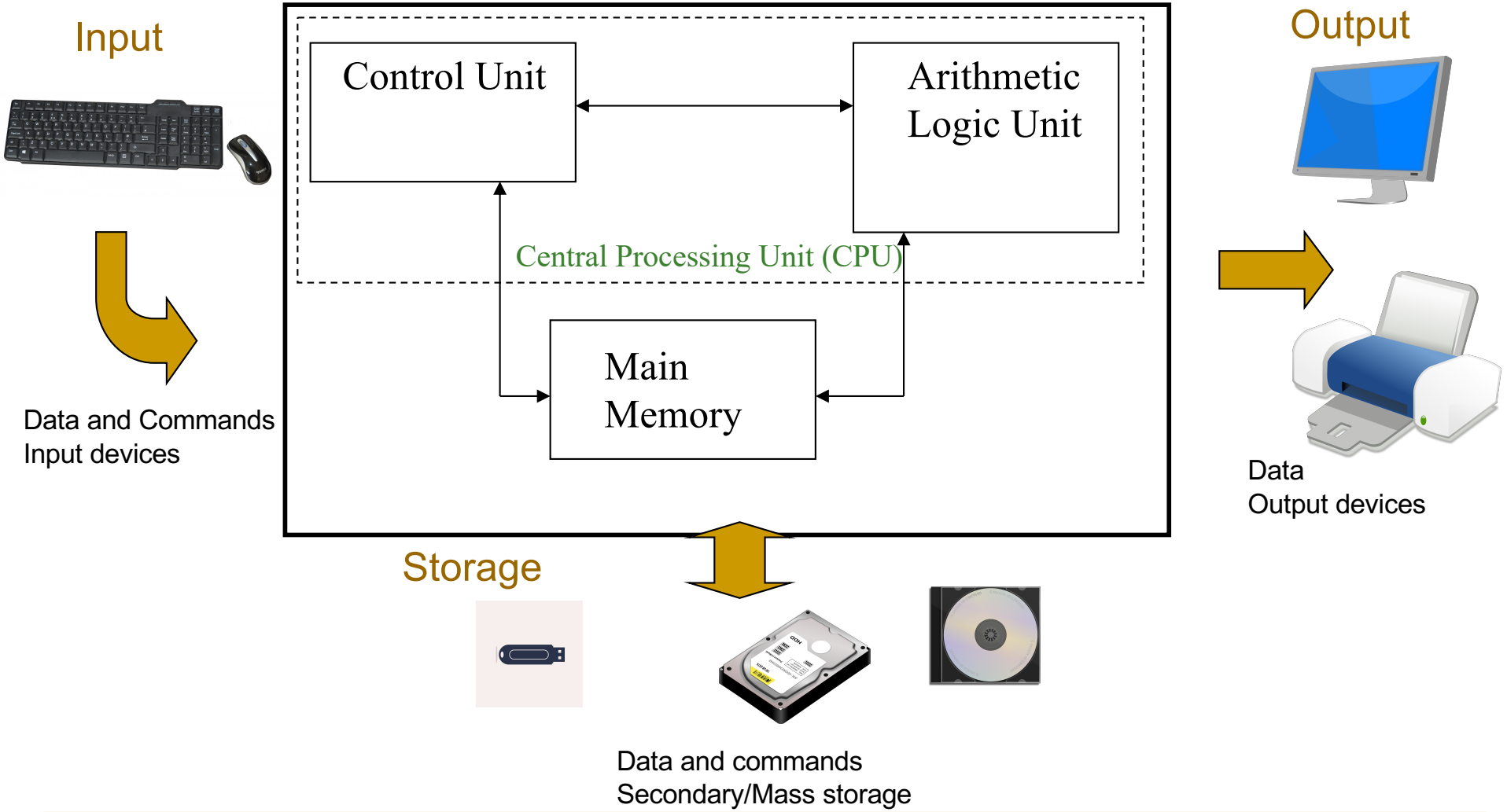
Output



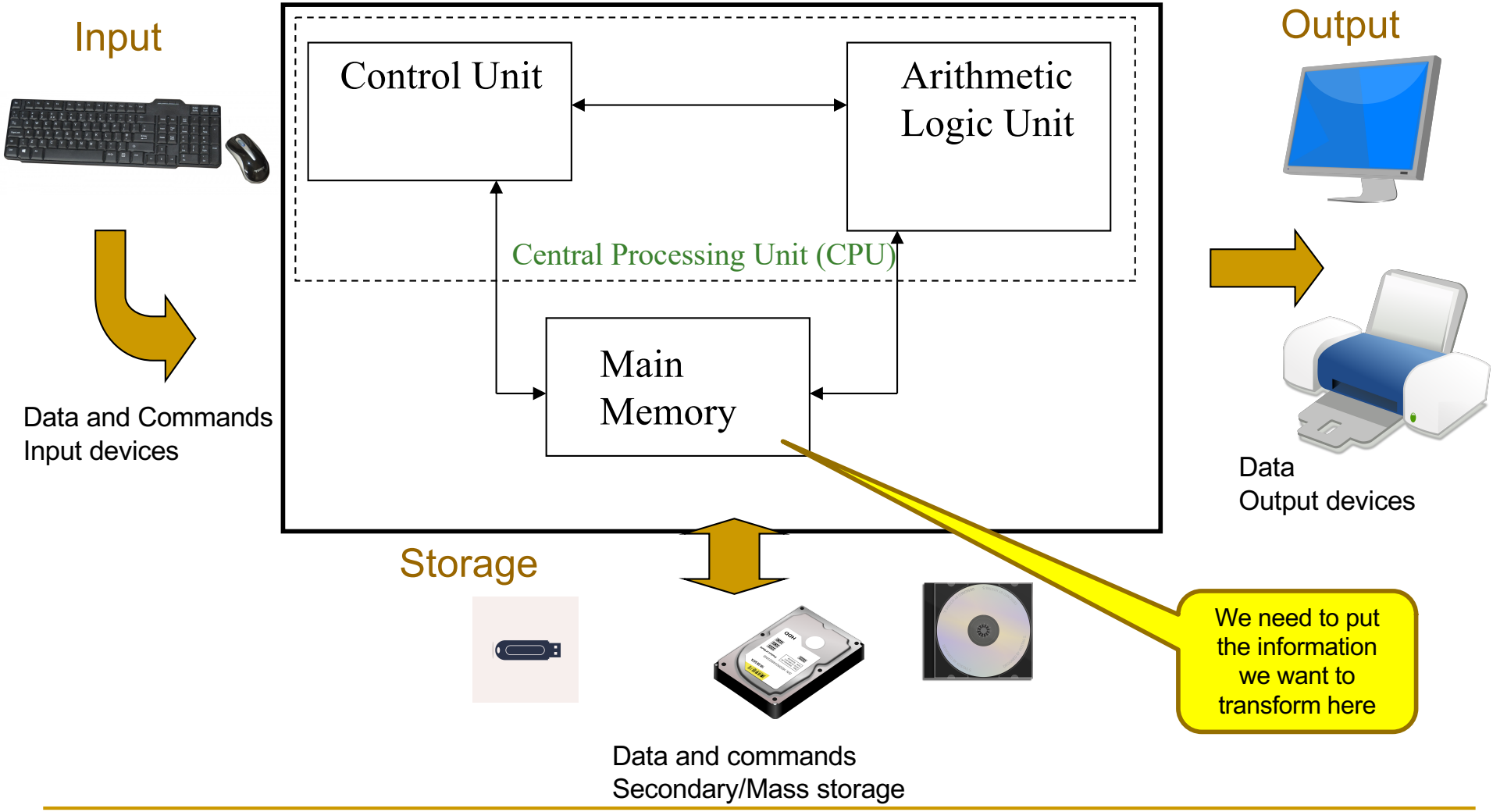


Storage

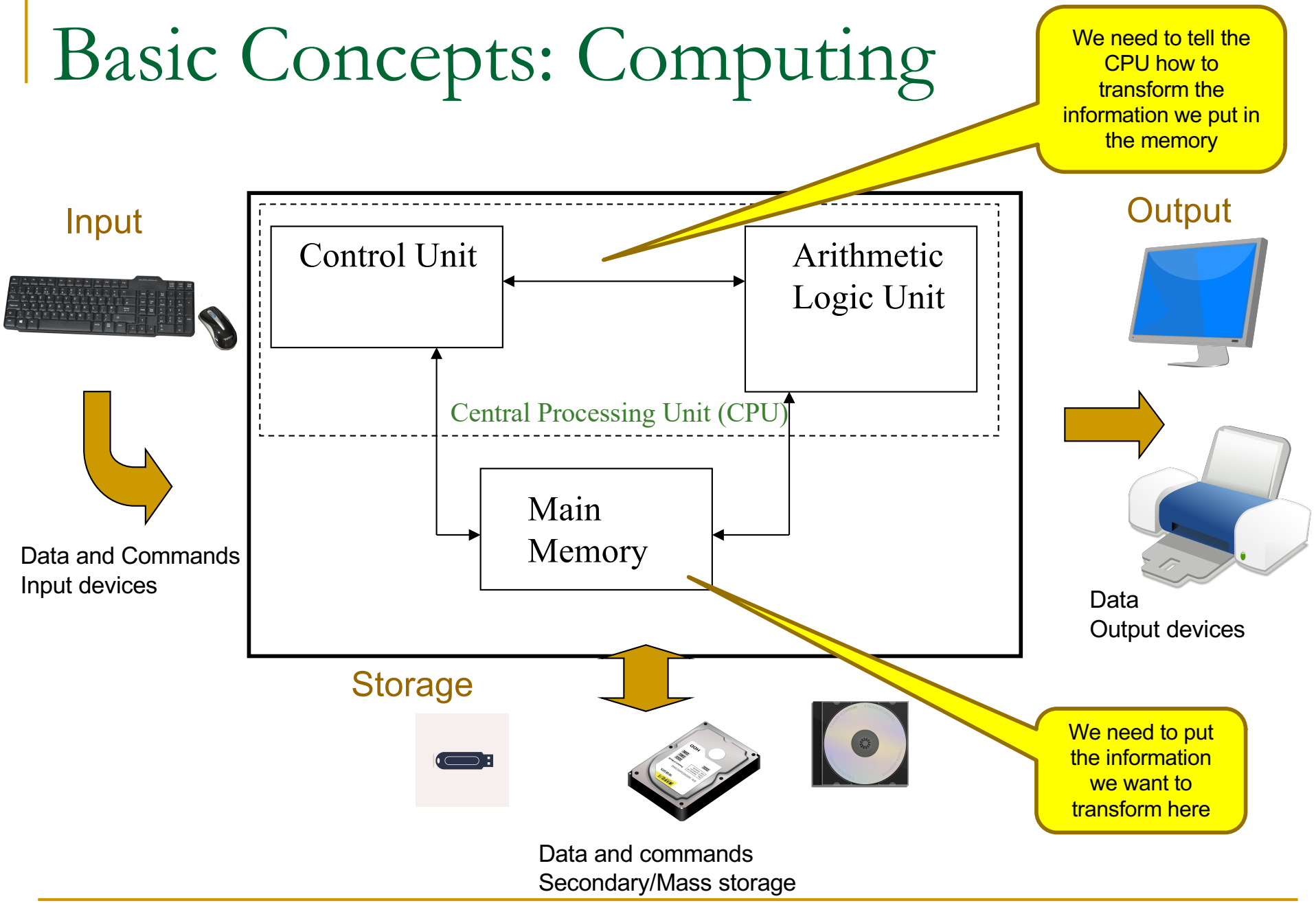
Basic Concepts: Computing



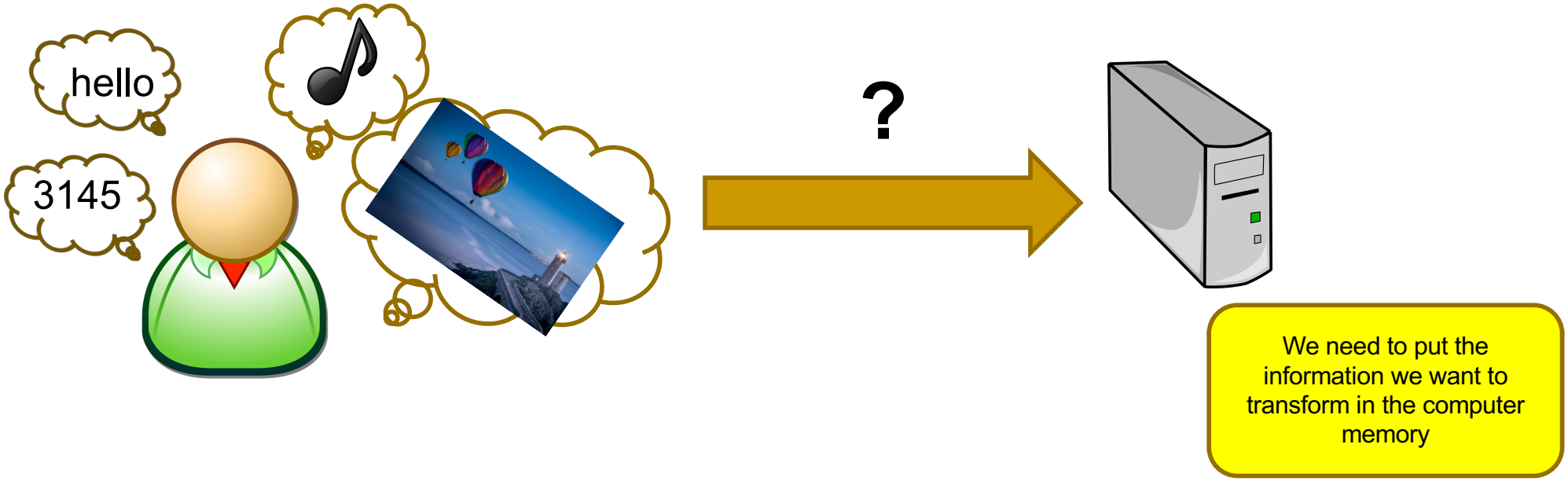
Basic Concepts: Computing



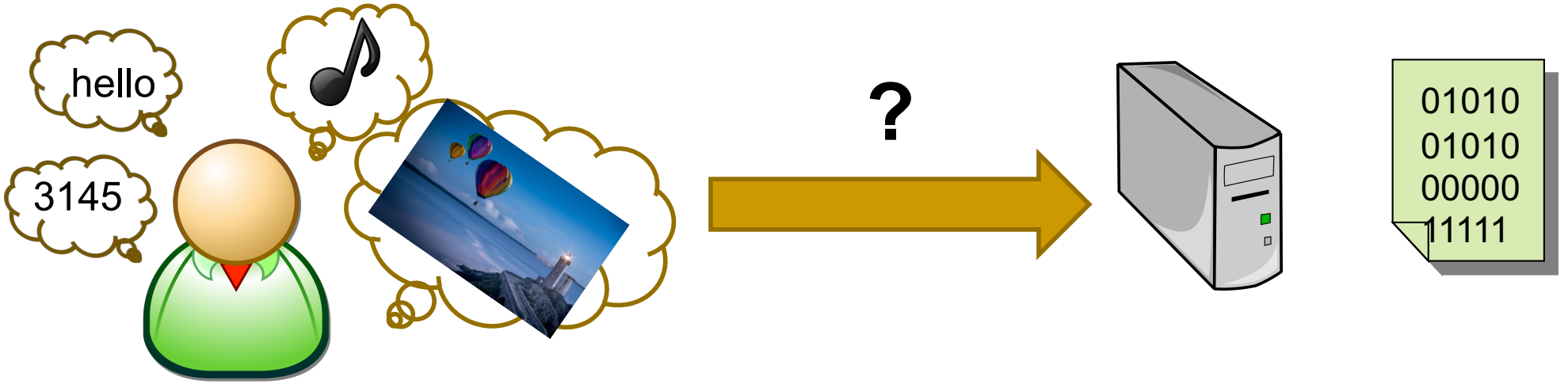
Basic Concepts: Computing



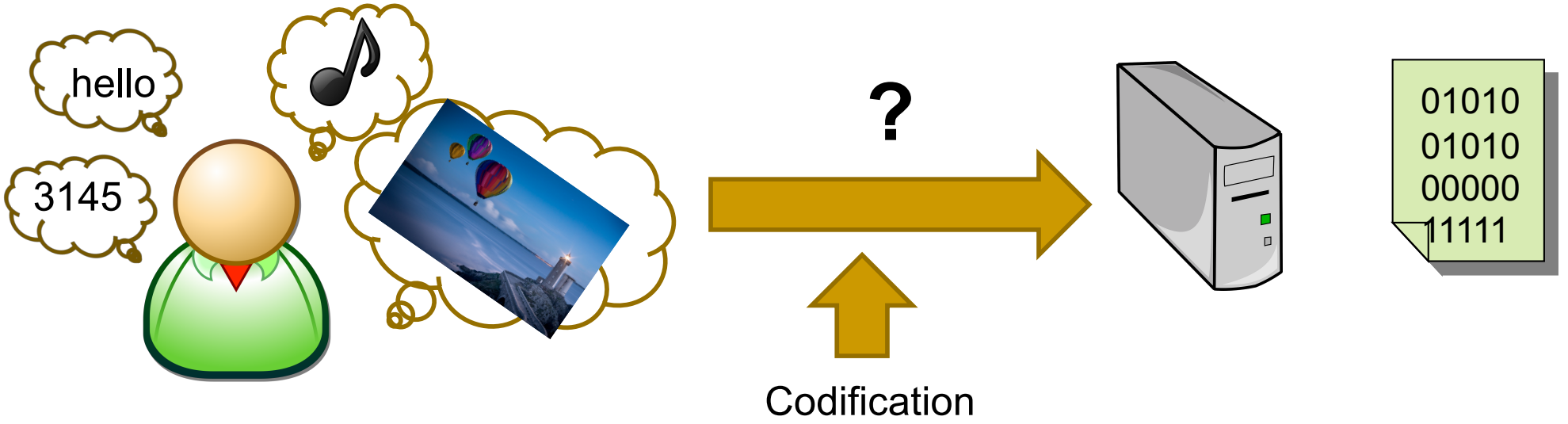
Basic Concepts: Information Representation



Basic Concepts: Information Representation



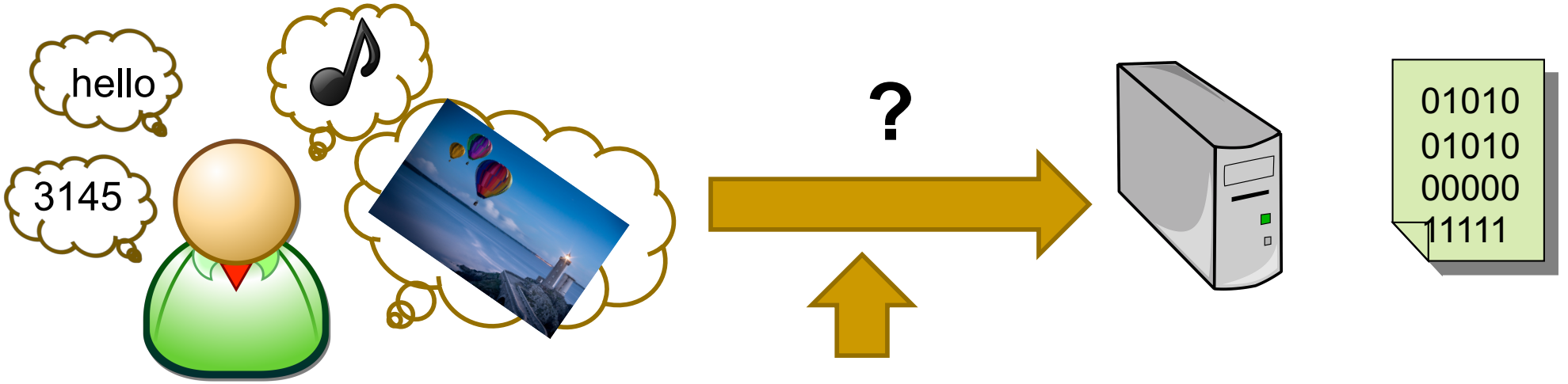
Basic Concepts: Information Representation



Information Representation

- Information and data managed by a computer are:
  - Elementary units: bit (Binary digIT)
  - Two possible states
  - Are represented by 0 and 1
- Bits can be used to represent characters, numbers, commands, code, colors, …
- 8 bits = 1 byte (256 different values)
- Kilobyte (Kb): 1000 Bytes.
- Megabyte (Mb): 1000 Kb.
- Gigabyte (Gb): 1000 Mb

Basic Concepts: Information Representation



Codification

Numeric Data	Integer, Real
Text	ASCII, Unicode
Sound	Wav, MIDI, Mp3
Images	Bitmap, Vectorial

Information Representation

- Representing **numbers**:
  - *Decimal numeral system*
    - 10 digits
  - *Binary numeral system*
    - 2 digits
  - *Hexadecimal numeral system*
    - 16 digits

There are methods and formulas to directly convert a decimal number into a binary one and viceversa

Decimal	Binary	Hexadecimal
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

Information Representation

- Representing **numbers**:
  - Positive and negative numbers
    - Absolute value and sign
      - **0**11 = 3 **1**11 = -3

    If the first digit is 0 it stands for +,
    if the first digit is 1 it stands for -

  - Real
    - Floating
      - $-324.8125_{(10} = 101000100.1101_{(2}$

sign	exponent	mantissa

Decimal	Binary	Hexadecimal
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

Information Representation

- Representing **characters**
  - Each character included in the keyboard is represented by binary numbers
    - ASCII
      - extended
      - Fixed coding (1 character = 1 byte)
    - Unicode
      - Unique template for 65.000 characters
      - Language independent
      - Variable coding (1 or 2 bytes)
      - Used on the web very often

ASCII		
Character	**Decimal**	**Binary**
A	65	0100 0001
B	66	0100 0010
. . .		
Z	90	0101 1010
a	97	0110 0001
z	122	0111 1010

Note that the relation between the character and the number used to codify is **totally arbitrary**

Programming Languages

We need to tell the CPU how to transform the information we put in the memory

- What is Programming?
  - A list of commands (sequence):
    - Executable by the CPU
    - Perform a certain task
  - Phases:
    1. Problem solving => creating an algorithm
    2. Adapting the algorithm to the computer => codify the algorithm by using a language understandable by the computer

Programming Languages

- **Algorithm Example:**

Problem:

Compute the lenght of a vector.

Input data:

(x1, y1), (x2, y2)

Algorithm:

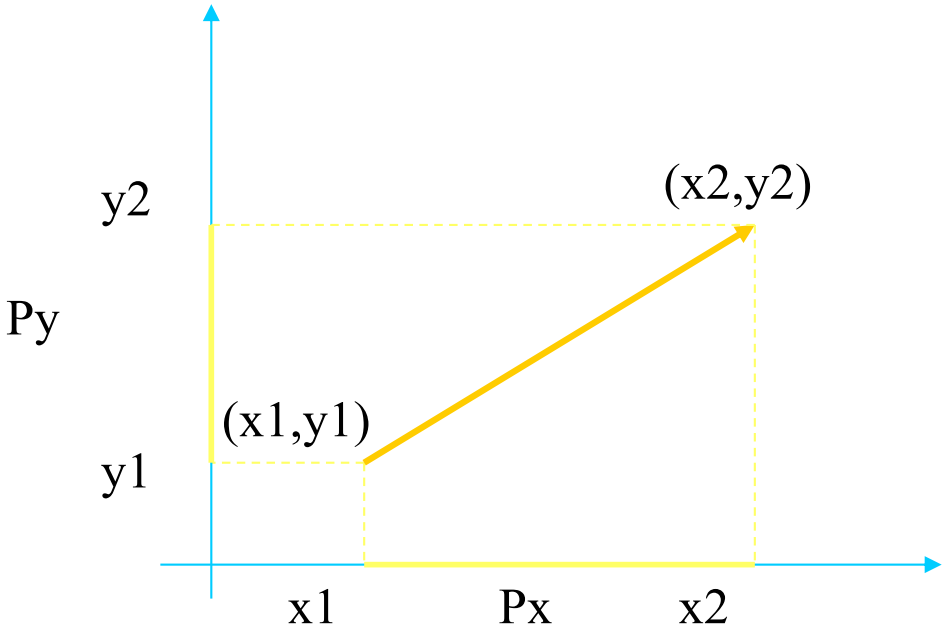
- Read x1, y1, x2, y2
- Compute Px = x2 – x1
- Compute Py = y2 – y1
- Compute $Px^2 = Px \cdot Px$
- Compute $Py^2 = Py \cdot Py$
- Compute R = sqrt2 ($Px^2 + Py^2$)
- Print R

Output data:

R

Program :

Algorithm steps detailed using Matlab.



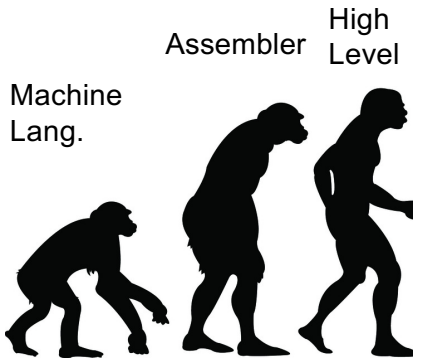
Programming Languages

- **Language:**
  - a set of symbols (characters, number, …) and rules allowing communication
- **Programming Language:**
  - a set of symbols (characters, number, …) and rules allowing communication between the developer and the computer.
  - Includes alphabet, syntax and semantics.
- **Algorithm:**
  - an algorithm is a finite list of well-defined instructions for accomplishing some task that, given an initial state, will proceed through a well-defined series of successive states, possibly eventually terminating in an end-state; it is not codified in a specific programming language
- **Program**:
  - is a collection of instructions that describes a task, or set of tasks, to be carried out by a computer in a specific **programming language**.
    - Program = data sets and algorithms

Programming Languages

- Levels:
  - **Low level or machine language:**
    - Binary language interpreted by CU
    - Hardware Specific
    - Commands include a operation code and operands
  - **Assembler**:
    - assembly instruction mnemonics into *operation codes*:

Machine Language	01000	001
Assembler	INC	CX

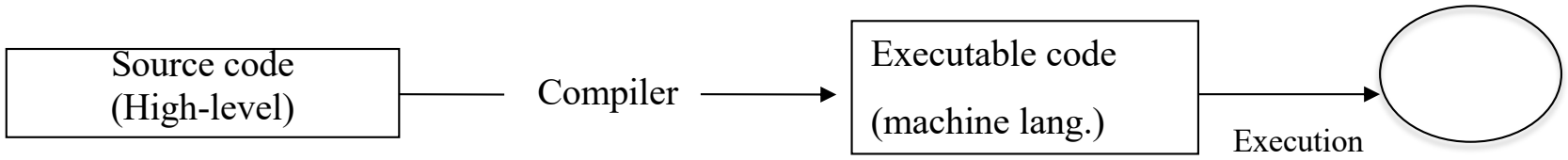


  - **High level.**
    - Commands are written in a language closer to the humans (*read*, *get*, *while*, integer, …).
    - Architecture independent.
    - High level of abstraction. Easy to develop.
    - Reduced program size.
    - Allow data-types (integers, real, characters, …)
    - **Need translation into machine code: compilation**

Compiled Languages

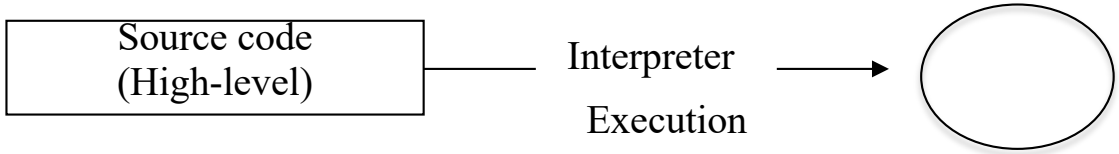
- They need special programs (compilers) which translate source code (written in high-level language) and **produce a separated file containing machine code**
  - Compiler:
    - Translate a complete program (source code) into object code (binary)
    - Object code is stored into memory and can be executed directly
    - During the translation task errors can be detected

To execute the program you will need the executable code



Interpreted languages

- The source code is also compiled to produce the machine code that the processor can execute but the process to do that is different from compiled languages
- Source code is translated sequentially one statement at a time using a **command interpreter**
  - They do not produce a separate executable file when compiled by the command interpreter
  - To execute the program you will need the source code and the command interpreter
  - Interpreted languages are usually slower as programs need to be compiled each time we want to execute them
  - In other words: **the source code is compiled each time you execute the program**



Types of Programming Languages

- Classification based on the **type of compilation:**
  - Compiled Programs: C, C++, Java, Basic
  - Interpreted Programs: MATLAB, Javascript

- Classification based on the **purpose of the language:**
  - General purpose (Basic, Pascal, C)
  - Problem oriented (Fortran, Cobol, Lisp, SQL)

- Classification based on the **programming paradigm**:
  - Imperative programming: Pascal, C, MATLAB
  - Object Oriented Programming: Java, C++
  - Functional programming: Lisp
  - Logic programming: Prolog

Bibliography

- Handbook of theoretical computer science (vol. B): formal models and semantics, EA Emerson - 1991 - MIT Press, Cambridge, MA